

SDEV1201 Database Fundamentals

LESSON 12

Let's review

Entity Relationship Model

Entity

An entity is a person, place, thing or concept that is used in the business context of the application and for which data is collected.

Attributes

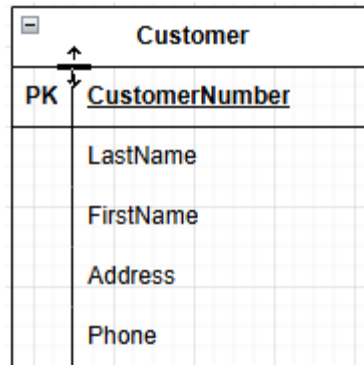
A piece of data that describes an entity is called an **attribute**. Synonyms for attribute include: element, property and field. In the order example, the customer entity has the following attributes:

- Customer Number
- Last Name
- First Name
- Address
- Phone Number

The values for all attributes describing an entity make up one instance of the entity. For example, the following values describe one specific customer.

Customer Number	Last Name	First Name	Address	Phone
100	Adams	Joan	9825-77 Ave Edm Ab T5C 8E2	488-8712

Entities and Tables



After the ERD design is complete, when the database is created the entities become tables.

<u>CustomerNumber</u>	FirstName	<u>LastName</u>	Address	Phone
100	Adams	Julie	9825-77 Ave Edmonton AB T5C 8E2	780-111-1111
200	Smith	Miles	4308-133 Ave Edmonton AB T5C 8E9	780-222-2222

What Are Data Types?

Unlike humans, a computer does not know the difference between "1234" and "abcd." A data type is a classification that dictates what a variable or object can hold in computer programming. Data types are an important factor in virtually all computer programming languages. Each data type requires different amounts of memory and has specific operations that can be performed over it.

CREATE TABLE **minimum syntax**

11

```
CREATE TABLE TableName (  
    Column1 DATATYPE  
    , Column2 DATATYPE  
    , Column3 DATATYPE  
    ...  
)
```

where *Column#* represents the name of the attribute/column and *DATATYPE* is *INT*, *DATETIME*, etc.

Constraints are used to:

21

1. Define the **PK**
 - This is defined by the **primary key** constraint
2. Define **relationships**
 - This is defined by the **foreign key** constraint
3. Define **default values**
 - Defined by the **default** constraint
4. Define a **domain of valid values**
 - Defined by the **check** constraint
5. Ensure that **all values in a column are unique**

Primary Key Constraint

When using a normalized database design, all tables must have a primary key constraint. Any column that acts as a primary key must be defined as a not null column. The DBMS will ensure that all rows in the table have a unique value for primary key columns.

When a primary key constraint is defined, PostgreSQL automatically creates a unique index for the column(s) acting as the primary key.

Syntax

Column constraint syntax:

[Constraint constraint_name] Primary Key

Table constraint syntax:

[Constraint constraint_name]

Primary Key (col_name [, col_name2 [, . . . col_name32]])

FK Constraints & referential integrity

The foreign key constraint defines referential integrity between two tables. Operations that affect the tables and/or the data stored within the tables must comply with referential integrity. This affects:

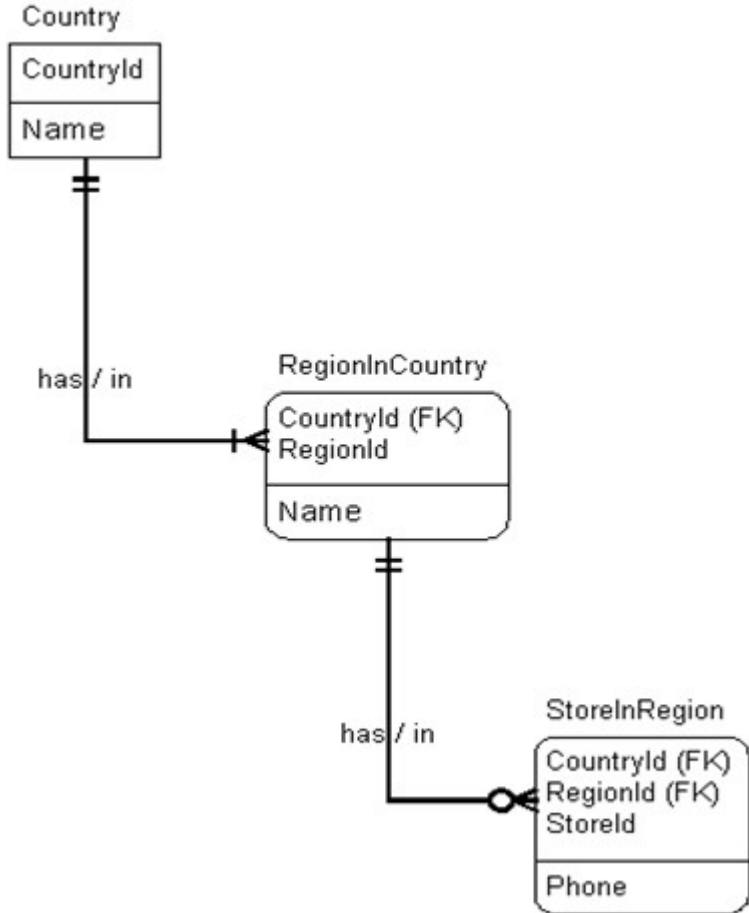
1. Dropping tables.
2. Creating tables.
3. Inserting/Updating/Deleting rows in tables.

The bottom line for referential integrity is that a child row/table cannot exist without its associated parent row/table. Therefore:

1. You must Drop a child table BEFORE you Drop the associated parent table.
2. You must Create a parent table BEFORE you Create the associated child table.
3. The value of a column acting as a foreign key must be:
 - A value that exists as a primary key in the associated parent table, or
 - NULL

For example

31



Drop

1. DROP TABLE StoreInRegion
2. DROP TABLE RegionInCountry
3. DROP TABLE Country

Create

4. CREATE TABLE Country (...)
5. CREATE TABLE RegionInCountry (...)
6. CREATE TABLE StoreInRegion (...)

Foreign Key Constraint Syntax

Column constraint syntax:

[Constraint constraint_name]
References ref_table (ref_column)

Table constraint syntax:

[Constraint constraint_name]
Foreign Key (column_name [, . . . column_name32])
References ref_table (ref_column [, . . . ref_column32])

- column_name identifies the column(s) acting as the foreign key
- ref_table identifies the parent table
- ref_column identifies the primary key in the associated parent table

Indexes

An index is an object stored in the db.

It's associated with 1 or more columns, in a specific table, that act as **the key for the index**.

This helps the DBMS look up (retrieve) the data quicker.

Functionally an index object is similar to an index at the back of a book.

Syntax

```
Create [Unique] Index [If Not Exists] index_name  
    On table_name ( column_name [, ...n] [ Asc | Desc] );
```

Column name represents the key for the index. One or more columns may act as the key for the index. All indexes must have a unique name. In SDEV1201 we normally use the prefix **IX** when naming an index.

Question 2 Solution for “01 Create Table PostgreSQL

Create Table Vehicles

```
(  
  VehicleNumber integer Not Null Constraint PK_Vehicles_VehicleNumber Primary Key,  
  DriverFirstName varchar (35) Not Null,  
  DriverLastName varchar (35) Not Null  
);
```

Create Table Deliveries

```
(  
  DeliveryNumber integer Not Null Constraint PK_Deliveries_DeliveryNumber Primary Key,  
  DeliveryDate date Not Null,  
  VehicleNumber integer Not Null  
                                Constraint FK_Deliveries_VehicleNumber_to_Vehicles_VehicleNumber  
                                References Vehicles (VehicleNumber)  
);
```

Question 2 Solution for “01 Create Table PostgreSQL

Create Table Customers

```
(  
  CustomerNumber integer Not Null Constraint PK_Customers_CustomerNumber Primary Key,  
  FirstName varchar (35) Not Null,  
  LastName varchar (35) Not Null,  
  StreetAddress varchar (35) Not Null,  
  City varchar (35) Not Null,  
  Province char (2) Not Null,  
  PostalCode char (6) Not Null,  
  PhoneNumber char (12) Not Null  
);
```

Create Table Products

```
(  
  ProductNumber char (7) Not Null Constraint PK_Products_ProductNumber Primary Key,  
  Description varchar (50) Not Null,  
  CurrentPrice decimal (7,2) Not Null  
);
```


Question 2 Solution for “01 Create Table PostgreSQL

Create Table Orders

```
(  
  OrderNumber integer Not Null Constraint PK_Orders_OrderNumber Primary Key,  
  OrderDate date Not Null,  
  FloristNumber integer Not Null,  
  CustomerNumber integer Not Null  
      Constraint FK_Orders_CustomerNumber_To_Customers_CustomerNumber  
      References Customers (CustomerNumber),  
  GSTAmount decimal (7,2) Not Null,  
  DeliveryNumber integer Not Null  
      Constraint FK_Orders_DeliveryNumber_To_Deliveries_DeliveryNumber  
      References Deliveries (DeliveryNumber),  
  DeliveredYN char (1) Not Null,  
  PaidYN char (1) Not Null  
);
```

Question 2 Solution for “01 Create Table PostgreSQL

Create Table OrderDetails

```
(  
    OrderNumber integer Not Null  
        Constraint FK_OrderDetails_OrderNumber_To_Orders_OrderNumber  
        References Orders (OrderNumber),  
    ProductNumber char (7) Not Null  
        Constraint FK_OrderDetails_ProductNumber_To_Products_ProductNumber  
        References Products (ProductNumber),  
    Quantity integer Not Null,  
    SellingPrice decimal (7,2) Not Null,  
    Constraint PK_OrderDetails_OrderNumber_ProductNumber  
        Primary Key (OrderNumber, ProductNumber)  
);
```

Question 2 Solution for “01 Create Table PostgreSQL

-- Indexes

-- Create indexes on all foreign keys.

Create Index IX_Deliveries_VehicleNumber On Deliveries (VehicleNumber);

Create Index IX_Orders_CustomerNumber On Orders (CustomerNumber);

Create Index IX_Orders_DeliveryNumber On Orders (DeliveryNumber);

Create Index IX_OrderDetails_OrderNumber On OrderDetails (OrderNumber);

Create Index IX_OrderDetails_ProductNumber On OrderDetails
(ProductNumber);

Today's Objective

By the end of this section you will be able to understand:

- ❑ Default Constraint.
- ❑ Unique Constraint.
- ❑ Check Constraint.
- ❑ Alter Table.

Information on these topics can be found in Brightspace “Week 6 – Alter Table & Constraints” in the “04B Constraints DF UN CK PostgreSQL” and “05 Alter Table PostgreSQL” documents.

Default Constraint

Default Constraint

The default constraint lets you define a value that is assigned to a column when the user does not specifically supply a value when a new row is added to the table (using the Insert statement). A default constraint can be defined on any column except:

- A column with the serial property (not discussed in this course)

The value of the default can be supplied via a constant (5.90) or a function (i.e. `Current_Date`). The default constraint name should use a prefix of DF (indicating this is a default constraint).

Default Constraint

The value of the default can be supplied via a constant (5.90) or a function (i.e. Current_Date). The default constraint name should use a prefix of DF (indicating this is a default constraint).

Its name should use a prefix of DF.

Syntax:

[Constraint constraint_name]

Default {constant | function}

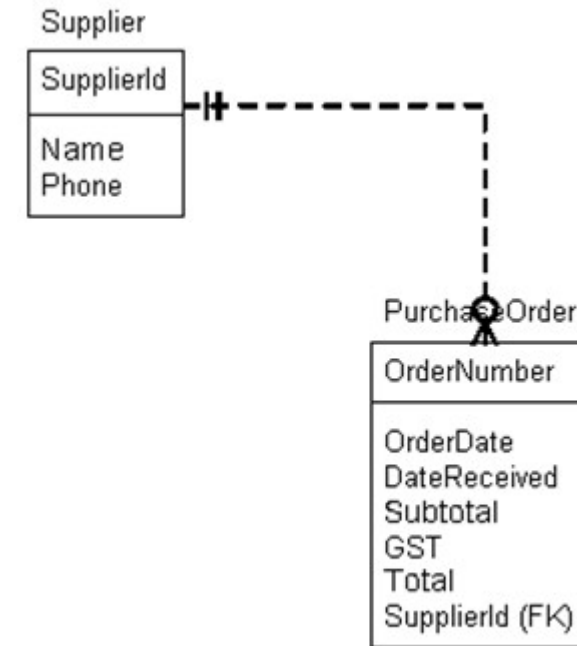
Default Constraint

Examples:

Default	Constraint Definition
Use the current date as the default for the DateReceived column	Constraint DF_DateReceived Default Current_Date
Use 5.90 as the default for the HourlyRate column	Constraint DF_HourlyRate Default 5.90

Example

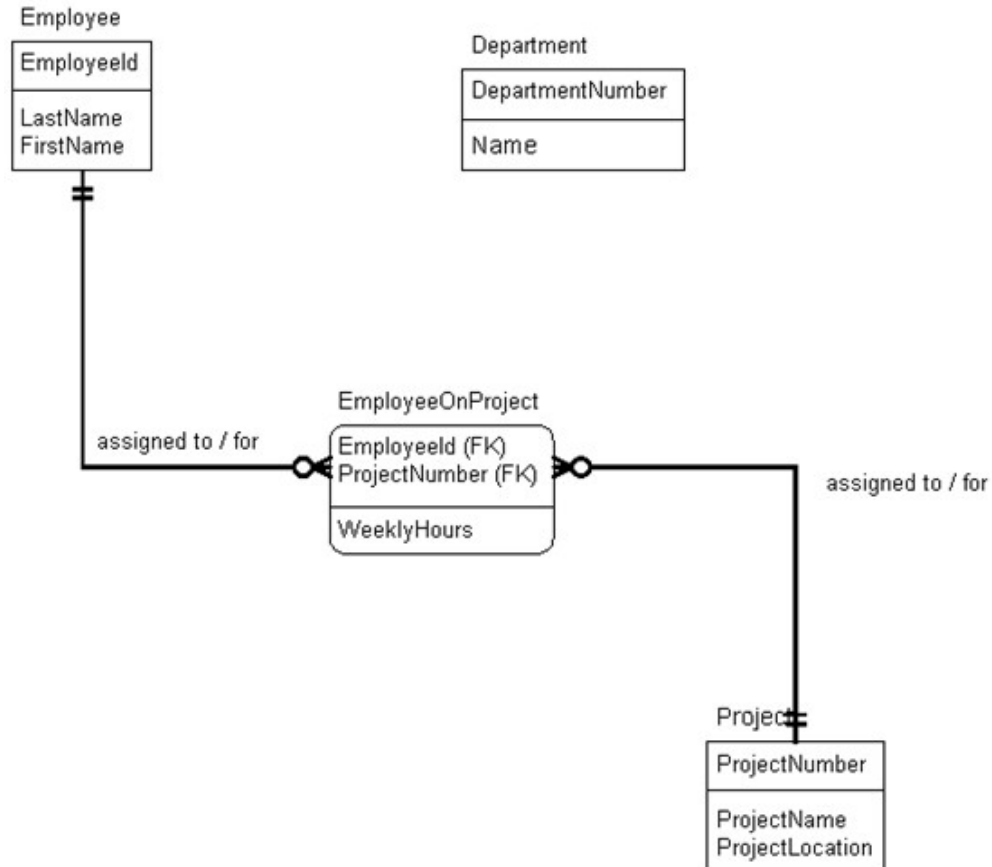
A column level definition is used to define a default constraint.



```
Create Table PurchaseOrder
(
    OrderNumber integer not null
        Constraint PK_PurchaseOrder Primary Key,
    OrderDate date not null
        Constraint DF_OrderDate
            Default Current_Date,
    DateReceived date not null,
    SupplierID integer not null
        Constraint FK_PurchaseOrderToSupplier References Supplier (SupplierID),
    SubTotal decimal (6,2) not null,
    GST decimal (4,2) not null,
    Total decimal (8,2) not null
);
```

Practice

Establish 5.0 as the default number of hours per week that an employee will work on a project.



```
CREATE TABLE EmployeeOnProject(  
    EmployeeID char(11) not null  
        CONSTRAINT FK_EmployeeOnProjectToEmployee  
        REFERENCES Employee (EmployeeID)  
    ,ProjectNumber int not null  
        CONSTRAINT FK_EmployeeOnProjectToProject  
        REFERENCES Project (ProjectNumber)  
    ,WeeklyHours int not null  
        CONSTRAINT CK_HoursLimit CHECK (WeeklyHours <= 20)  
    ,CONSTRAINT PK_EmployeeOnProject PRIMARY KEY (EmployeeID, ProjectNumber)  
)
```

Solution

```
CREATE TABLE Employee (  
    EmployeeID char(11) not null CONSTRAINT PK_Employee PRIMARY KEY  
    ,Lastname varchar(100) not null  
    ,Firstname varchar(100) not null  
)
```

```
CREATE TABLE Project (  
    ProjectNumber int identity(1, 1) not null CONSTRAINT PK_Project PRIMARY KEY  
    ,ProjectName varchar(100) not null  
    ,ProjectLocation varchar(100) not null  
)
```

```
CREATE TABLE EmployeeOnProject(  
    EmployeeID char(11) not null  
        CONSTRAINT FK_EmployeeOnProjectToEmployee  
        REFERENCES Employee (EmployeeID)  
    ,ProjectNumber int not null  
        CONSTRAINT FK_EmployeeOnProjectToProject  
        REFERENCES Project (ProjectNumber)  
    ,WeeklyHours int not null  
        CONSTRAINT DF_Hours DEFAULT 5  
        CONSTRAINT CK_HoursLimit CHECK (WeeklyHours <= 20)  
    ,CONSTRAINT PK_EmployeeOnProject PRIMARY KEY (EmployeeID, ProjectNumber)  
)
```

```
CREATE TABLE Department (  
    DepartmentNumber int identity(1, 1) not null  
    ,Name varchar(50)  
)
```

Unique Constraint

Unique Constraint

The unique constraint makes sure that the values in one or more columns are not repeated. When applied to a single column, this ensures all entries in that column are different. When applied to multiple columns, this requires that the combination of values across those columns is unique, so no two rows can have exactly the same values.

A unique constraint does not consider null values to be equal.

Unique Constraint

Adding a unique constraint will automatically create a unique index on the column or group of columns used in the constraint.

Column Constraint Syntax

`[Constraint constraint_name] Unique`

Table Constraint Syntax

`[Constraint constraint_name]`

`Unique (col_name [, col_name2 [ , . . . col_name32] ] )`

Unique Constraint

Students cannot use the same Email address, and multiple students cannot have the same first and last names.

Create Table Student

```
(  
    StudentID integer not null,  
    FirstName varchar (30) not null,  
    LastName varchar (30) not null,  
    Email varchar (50) not null  
    Constraint UQ_Student_Email Unique,  
    Constraint UQ_Student_Names Unique (FirstName, LastName)  
);
```

Check Constraint

Check Constraint

The **CHECK** constraint enables you to specify what values are acceptable (i.e. define a DOMAIN).

CHECK constraints should be given a meaningful name and consist of an expression that evaluates to TRUE or FALSE.

Syntax:

```
CONSTRAINT CK_ConstraintName CHECK (expression)
```

Whenever a new row of data is added to the table, or an existing column has its value changed the new data will be evaluated using the expression in the check constraint. If the expression evaluates to true the DBMS will add the new data to the table. If the expression evaluates to false an error message is issued and the new data is rejected.

Check Constraint

The check constraint name should begin with the prefix CK (identifies the constraint as a check constraint) and be meaningful. Constraint names must be unique within the database. The syntax is identical whether the constraint is defined at the column level or the table level.

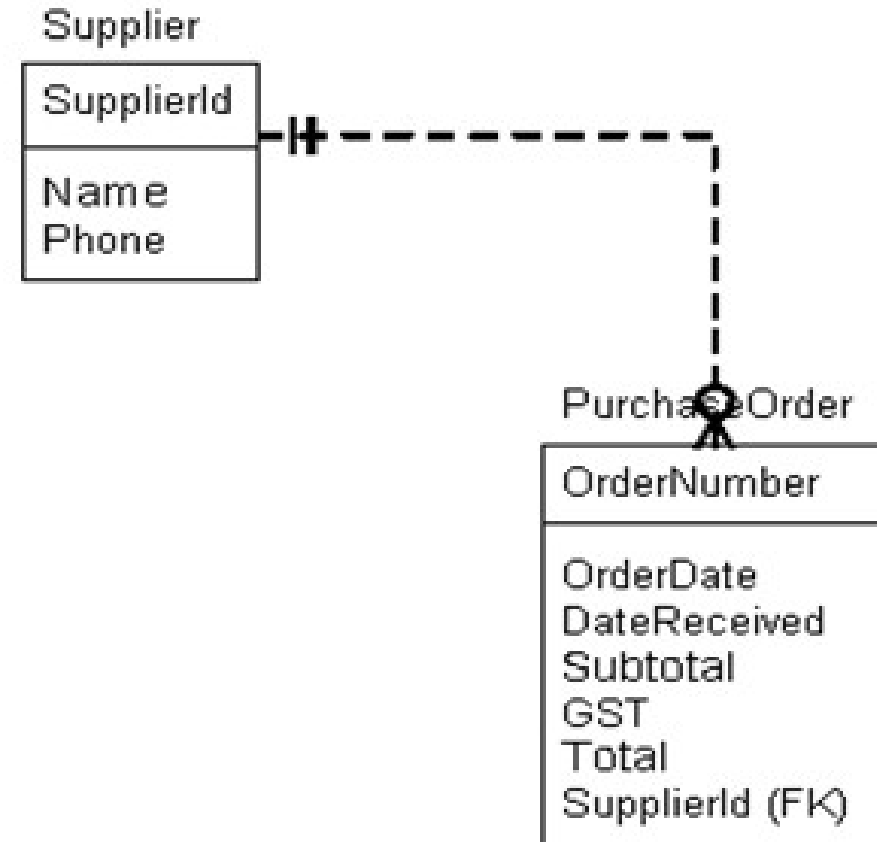
The expression in the check constraint:

- cannot contain a subquery
- must evaluate to true or false
- can be a compound Boolean expression
- can reference another column in the same table if the constraint is a table constraint

Check Constraint Examples

Column	Domain	Check Constraint
QuantitySold	Must be positive	Constraint CK_QuantitySold Check (QuantitySold > 0)
DateReceived	Must be >= the Date Ordered	Constraint CK_DateReceived Check (DateReceived >= DateOrdered)
CourseMark	Must be >= 0 and <= 100	Constraint CK_CourseMark Check (CourseMark Between 0 and 100) Or Constraint CK_CourseMark Check (CourseMark >= 0 and CourseMark <= 100)
PostalCode	Must follow the pattern A9A 9A9	Constraint CK_PostalCode Check (PostalCode ~ '[A-Z][0-9][A-Z] [0-9][A-Z][0-9]')

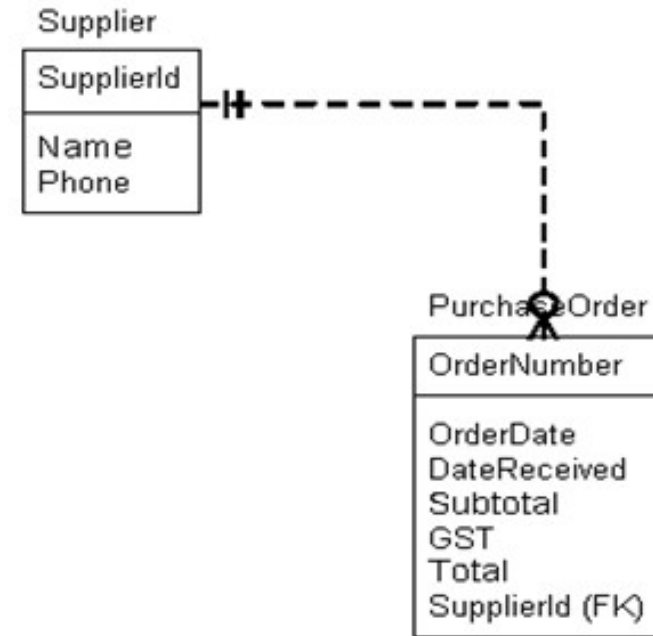
Example



```
Create Table Supplier
(
    SupplierID integer not null
    Constraint PK_Supplier Primary Key,
    Name varchar (100) not null,
    Phone char (14) not null
    Constraint CK_Phone
    Check (Phone ~ '\([0-9][0-9][0-9]\) [0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]')
);
```

Note: Left and right brackets are “special” characters, so they need to be escaped with a slash

Example



```
Create Table PurchaseOrder
(
    OrderNumber integer not null
        Constraint PK_PurchaseOrder Primary Key,
    OrderDate date not null,
    DateReceived date not null,
    SupplierID integer not null
        Constraint FK_PurchaseOrder_Supplier References Supplier (SupplierID),
    SubTotal decimal (6,2) not null
        Constraint CK_SubTotalMustBePositive Check (Subtotal > 0),
    GST decimal (4,2) not null
        Constraint CK_GSTMustBePositive Check (GST > 0),
    Total decimal (8,2) not null
        Constraint CK_TotalMustBePositive Check (Total > 0),
    Constraint CK_DateReceivedMustBeOnOrAfterOrderDate Check (DateReceived >= OrderDate)
);
```

Check Constraints

Note that the check constraint for DateReceived and OrderDate is defined as a table level constraint (because it involves two columns) while all other check constraints are defined at the column level.

To test a check constraint:

1. Use pgAdmin to check the definition of the table. This ensures the constraint definition is in place and is correct.
2. Add a row to the table using data that violates the constraint. You should receive an error message, and the row should not be added to the table. This will be covered in a later lesson.

ALTER TABLE

ALTER TABLE

- The **ALTER TABLE** statement is used to:
- Add a column (with or without constraints)
- Drop an existing column (assuming you do not break any referential integrity such as PK - > FK)
- Change the datatype of an existing column
- Add a default value to an existing column
- Drop a default from an existing column
- Add/Drop a Not Null constraint to an existing column
- Drop an existing constraint
- Add a table constraint
- Rename a column, or column constraint, or table

Alter Table table_name

{

Add [Column] column_name data_type [column_constraint [...]]

|

Drop [Column] column_name

|

Alter [Column] column_name Set Data Type data_type

|

Alter [Column] column_name Set Default expression

|

Alter [Column] column_name Drop Default

|

Alter [Column] column_name { Set | Drop } Not Null

|

Drop Constraint constraint_name

|

Add table_constraint

} [, ...]

Syntax

Alter Table

Note: column_constraint and table_constraint syntax are the same syntax as in the Create Table document and will be fully taught in a later lesson.

Example:

To add a column YearFounded to the table Club:

[Alter Table Club](#)

[Add Column YearFounded integer;](#)

To drop the column YearFounded from the table Club:

[Alter Table Club](#)

[Drop Column YearFounded;](#)

To change the datatype of CourseHours in the Course table from smallint to integer:

[Alter Table Course](#)

[Alter Column CourseHours Set Data Type integer;](#)

Simplified syntax to Rename a column

Alter Table table_name Rename [Column] old_column_name To new_column_name

Example:

To rename the LoginID column to UserName in the Staff table:

Alter Table Staff

Rename Column LoginID To UserName;

Simplified syntax to Rename a constraint

Alter Table table_name Rename Constraint old_constraint_name
To new_constraint_name

Example:

To rename the FK_Staff_PositionID_To_Position_PositionID
constraint on the Staff table to FK_Staff_PosID_To_Position_PosID :

Alter Table Staff

Rename Constraint FK_Staff_PositionID_To_Position_PositionID
To FK_Staff_PosID_To_Position_PosID;

Simplified syntax to Rename a table

Alter Table old_table_name Rename To new_table_name

Example:

To rename the Position table to JobPosition:

```
Alter Table Position  
    Rename To JobPosition;
```

Practice

Work on question 3 on “01 Create Table PostgreSQL” (found in “Week 6 – Alter Table & Constraints” in Brightspace)